MEMS 传感器中的噪声分析和识别，Allan，时间，Hadamard，重叠，修正，总体方差

作者：Andrea Vitali

主要元件	
LSM6DS3H	iNEMO 惯性模块：3D 加速度计和 3D 陀螺仪
LSM6DSM/LSM6DSL	iNEMO 惯性模块：3D 加速度计和 3D 陀螺仪
STEVAL-STLKT01V1	SensorTile 开发套件

目的和益处

本设计建议阐释了如何分析和识别 MEMS 传感器中的噪声。解释了 Allan 和 Hadamard 方差及其变体（重叠、修正和总体）。还提到了理论方差#1（Theo1）。

优势：

- 如何通过 Allan 和其他方差来表征 MEMS 传感器

信号和噪声

基本假设是：测量期间目标信号是恒定且平稳的。但是传感器输出是目标信号和噪声的总和。大体上，从长期看噪声应该平均为零。

测量期间采集多个样本。噪声的分析和识别可以帮助确定要使传感器输出的方差最小，需要对 多少个样本取平均。

标准方差的问题在于，它不能很好地表征增加数据运行长度时的情形。为了解决此问题，开发出了 Allan 方差。Allan 方差 σ^2 被计算为连续“样本”（2 样本方差）之间的平方差的平均值。通过在时间间隔 τ = m*Ts 中对 m 个样本求平均来计算“样本”，其中 Ts=1/Fs 是采样间隔，Fs 是采样频率。

Allan 方差等等

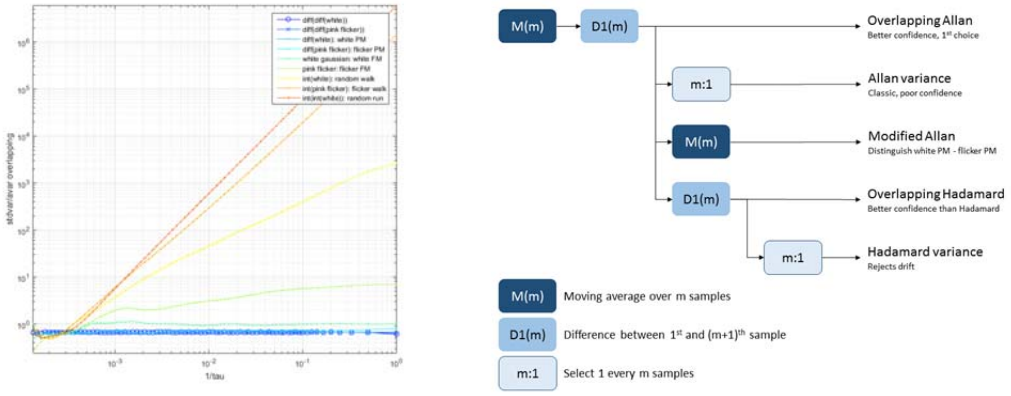
Allan 偏差 σ(τ)是 Allan 方差 $\sigma^2(\tau)$的平方根。对数-对数图的斜率取决于噪声类型。该斜率能够识别噪声类型。参见下表。

可以使用数字滤波器链来计算 Allan 方差（非重叠 AVAR，重叠 OAVAR，修正 MAVAR）和 Hadamard 方差（非重叠 HVAR 和重叠 OHVAR）：

- M(m)是 m 个样本的移动平均值，
- D1(m)是第 n 个样本与第(n+m+1) 个样本之间的一阶差，
- m：通过从 m 个样本中选择 1 个样本进行 1 次下采样

瞬态不得出现在输出中。通过对输出样本求平方和求平均来计算方差。偏差是方差的平方根。可以将偏差的置信度估计为偏差本身除以平均输出样本数的平方根。

Figure 1. 左：var/AVAR 之比；右：处理链路框图



一些重要说明：

- 重叠 Allan（OAVAR）和重叠 Hadamard 方差（OHVAR）具有更好的置信度，应该用于 m = 10%的数据运行长度情况下。
- 时间方差（TVAR）是修正 Allan 方差（MAVAR）的缩放版本，缩放因子是 $\tau^2/3$。对于白相位噪声它是最佳的（这意味着：白高斯噪声的导数，称为白频噪声）。
- Allan 总体、修正总体和 Hadamard 总体方差在 m = 30-50%的数据运行长度时具有更好的置信度。通过在两侧反射来扩展数据运行长度，以此来计算总体方差。对于 Allan，反射样本数量为 2m，对于修正 Allan 和 Hadamard，反应样本数量为 3m。
- 理论#1 方差（Theo1）在 m = 75%的数据运行长度下具有更好的置信度。对于所有其他方差，步幅时间与用于计算“样本”的平均时间相同，var(τ=m*Ts)。对于 Theo1，步幅时间是 m*Ts 减去平均时间；平均时间从 m/2 下降到 1；步幅时间从 m/2 降到 m-1；平均步幅时间为 0.75*m*Ts：Theo1(τ=0.75*m*Ts)。Theo1 有偏差，Theo1BR（去除偏差）是无偏的，m <10%时 Theo1H 是 Allan，而 m> 10%时是 Theo1BR。

用来计算方差的 MatLab 代码

这是将上述方差计算为数字滤波器链的参考 MatLab 代码。

```
% Allan 作为一个数字滤波器
n % 平均因子，平均 n 个样本
Fs % 采样频率
Ts=1/Ts;
tau=n*Ts;
```

```
%---- STDVAR 和 OSTDVAR
Mn=ones(1,n)/n; % 平均滤波器
dataM=filter(Mn,1,data); dataM=dataM(n:end); % 滤波并去除瞬态
ostdvar()=var(dataM);
stdvar()=var(dataM(1:n:end));

%---- AVAR 和 OAVAR
D1n=zeros(1,n+1); D1n(1)=+1; D1n(end)=-1; % 差分滤波器
dataMD=filter(D1n,1,dataM); dataMD=dataMD(n+1:end); % 滤波并去除瞬态
L=length(dataMD);          oavar()=0.5*sum(dataMD.^2)/(L);
L=length(dataMD(1:n:end));  avar()=0.5*sum(dataMD(1:n:end).^2)/(L);

%---- MAVAR
dataMDM=filter(Mn,1,dataMD);
dataMDM=dataMDM(n:end);
L=length(dataMDM);          mavar()=0.5*sum(dataMDM.^2)/(L);

%---- HVAR 和 OHVAR
dataMDD=filter(D1n,1,dataMD); dataMDD=dataMDD(n+1:end); %  滤波并去除瞬态
L=length(dataMDD);          ohvar()=0.5*sum(dataMDD.^2)/(L);
L=length(dataMDD(1:n:end)); hvar()=0.5*sum(dataMDD(1:n:end).^2)/(L);
```

上述代码中最慢的一行是移动平均滤波器。如果在添加新项的同时丢弃旧项并保持运行总量不变，则执行速度会快很多。

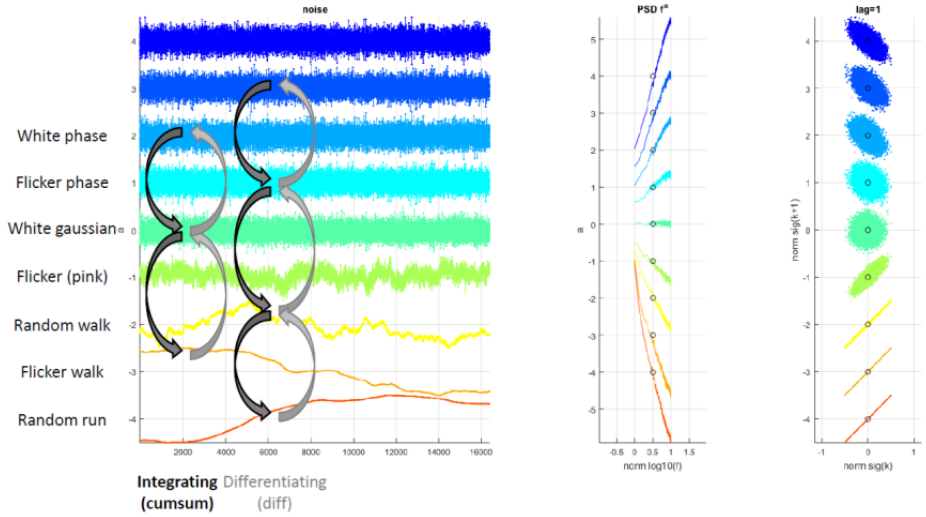
```
% OAVAR 最优化
n2=n*n;
acc(1)=sum(data(1:n)); % 初始化运行总数
for i=1:N-n, acc(i+1)=acc(i)-data(i)+data(i+n); end; % 运行总数
oavar()=0.5*sum( (acc(1:N-2*n+1) - acc(1+n:N-n+1)).^2 )/(N-2*n+1)/n2;

% AVAR
diffL=fix((N-2*n+1 -1)/n)+1;
avar()=0.5*sum( (acc(1:n:N-2*n+1)-acc(1+n:n:N-n+1)).^2 )/(diffL)/n2;

% STDVAR 和 OSTDVAR
stdvar()=var(acc(1:n:N-n+1))/n2;
ostdvar()=var(acc(1:N-n+1))/n2;

% MAVAR 最优化
n4=n2*n2;
acc2(1)=sum(acc(1:n)); % 初始化运行总数
for i=1:N-2*n+1, acc2(i+1)=acc2(i)-acc(i)+acc(i+n); end; % 运行总数
mavar()=0.5*sum( (acc2(1:N-3*n+2) - acc2(1+n:N-2*n+2)).^2 )/(N-3*n+2)/n4;

% OHVAR
ohvar()=0.5*sum( ( (acc(1  :N-3*n+1) - acc(1+  n:N-2*n+1)) - ...
                   (acc(1+n:N-2*n+1) - acc(1+2*n:N-  n+1)) ).^2 )/(N-3*n+1)/n2;

% HVAR
diffL=fix((N-3*n+1 -1)/n)+1;
hvar()=0.5*sum( ( (acc(1  :n:N-3*n+1) - acc(1+  n:n:N-2*n+1)) - ...
                  (acc(1+n:n:N-2*n+1) - acc(1+2*n:n:N-  n+1)) ).^2 )/(diffL)/n2;
```

代码可以进一步优化。例如，对于优化的 C 代码，有可能计算一次得出方差（但通常需要两次计算：第一次用于求均值，第二次用来求实际方差）。

噪声模型

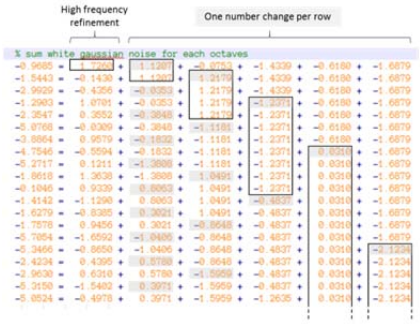
白高斯 (WH) 是第一种基本类型的噪声。通过积分（cumsum）可以得到 **Random Walk**（RW）。利用另一个积分，可以得到 **Random Run**（RR）。通过取导数（diff），可以从白频（WHFM）转移为白相噪声（WHPM）。

Figure 2. 噪声图，功率谱密度（PSD），滞后图（x_k vs x_{k+1}）



闪烁 1/f 噪声，也称为**粉红**噪声，是另一种基本类型的噪声。它等于求解 dx(t)/dt = n1*x(t) + n2*w(t)产生的噪声，其中 w(t)是白噪声。它可以通过对不同倍频程的不同白噪声发生器求和来生成。通过积分（cumsum），可以得到 **Flicker Walk**（FW）。通过取导数（diff），可以从闪烁频率（FLFM）转移为闪烁相位（FLPM）。

```
% 1/f 噪声发生器（闪烁，粉红）
  N % 噪声向量长度
  sd; % 噪声发生器的标准偏差
  n=zeros(1,N); % 初始化噪声向量
  imax=floor(log2(N)); ngen=randn(1,imax+1)*sd; % 初始化噪声生成
  ngensum=sum(ngen); % 初始化运行总数
  for i=1:N,
    % 在计数器中查找尾随零的数量
    tlz=0; t=i; while mod(t,2)==0, tlz=tlz+1; t=t/2; end;

    % 更新运行总数
    ngensum=ngensum-ngen(tlz+1); % 删除原数值
    ngen(tlz+1)=randn(1)*sd; % 更新噪声发生器
    ngensum=ngensum+ngen(tlz+1); % 添加新数值
```



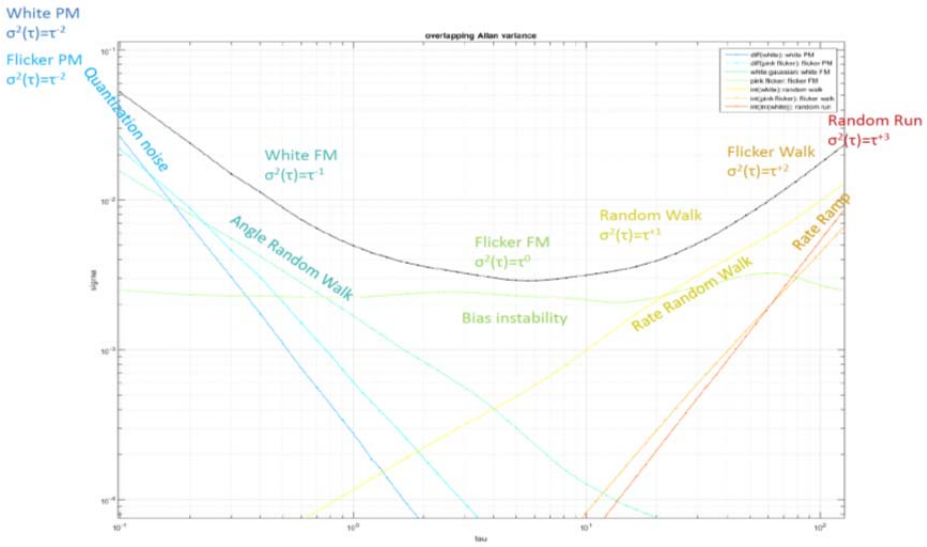
噪声识别

对于噪声识别，必须查看 sigma-tau 图（偏差，方差的平方根）和 sigma^2-tau 图（方差）的斜率。需要修正 Allan 或时间偏差/方差来区分白 PM 和闪烁 PM 噪声。

噪声类型	Matlab 代码	频谱 f^x	ADEV $\sigma(\tau)$	AVAR $\sigma^2(\tau)$	MADEV mod.$\sigma(\tau)$	MAVAR mod.$\sigma^2(\tau)$	TDEV $\sigma_T(\tau)$	TVAR $\sigma^2_T(\tau)$
白 PM	Diff(WhiteFM)	f^{+2}	τ^{-1}	τ^{-2}	$\tau^{-3/2}$	τ^{-3}	$\tau^{-1/2}$	τ^{-1}
闪烁 PM	Diff(FlickerFM)	f^{+1}	τ^{-1}	τ^{-2}	τ^{-1}	τ^{-2}	τ^{0}	τ^{0}
白 FM	Randn(1)	f^{0}	$\tau^{-1/2}$	τ^{-1}	$\tau^{-1/2}$	τ^{-1}	$\tau^{+1/2}$	τ^{+1}
闪烁 FM	参考以上内容	f^{-1}	τ^{0}	τ^{0}	τ^{0}	τ^{0}	τ^{+1}	τ^{+2}

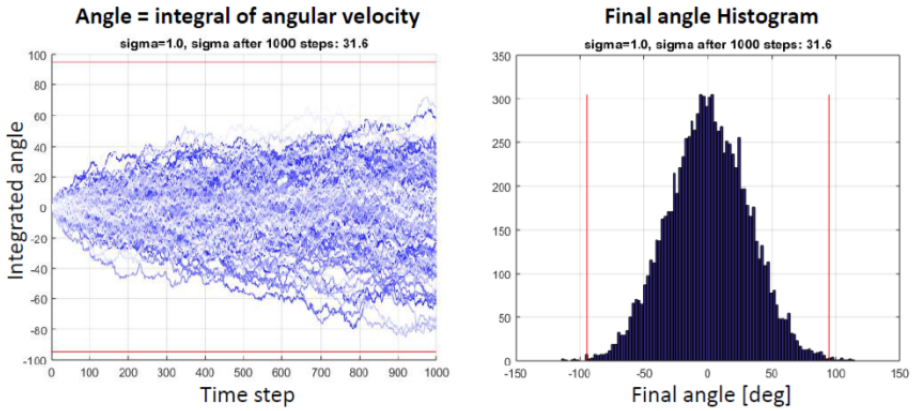
随机游走	Cumsum(WhiteFM)	f^{-2}	$\tau^{+1/2}$	τ^{+1}	$\tau^{+1/2}$	τ^{+1}	$\tau^{+3/2}$	τ^{+3}
闪烁游走	Cumsum(FlickerFM)	f^{-3}	τ^{+1}	τ^{+2}	τ^{+1}	τ^{+2}		
随机运行	Cumsum(RandomWal	f^{-4}	$\tau^{+3/2}$	τ^{+3}	$\tau^{+3/2}$	τ^{+3}		

Figure 3. Allan 方差图，$\sigma^2(\tau)$ vs τ，对应于具有特定斜率的不同类型噪声。



陀螺仪的噪声识别

陀螺仪输出是受白噪声影响的角速度。角位置通过积分获得。当陀螺仪不旋转时，输出应当是不为零的；而是具有零均值和给定标准差的白噪声。导致积分后会得到一个非零的积分角度。角度随机游走（ARW）。最终角度误差 RMS = ARW*sqrt(time)。示例：ARW 1deg/sqrt(s) * sqrt(1000s) = 31.6deg RMS。



角度随机游走可以在数据表中找到：**deg/sqrt(s)中的 ARW = 噪声密度 deg/s/sqrt(Hz)**。ARW 也可以在 Allan 图中找到：它是在 τ= 1s 处具有斜率 $\tau^{-1/2}$ 的 Allan 偏差段的截距，或是在 τ= 1s 处具有斜率 τ^{-1} 的 Allan 方差段的截距。

辅助资料

相关的设计支持材料
STEVAL-STLKT01V1，SensorTile 开发套件
文件
LSM6DS3H，iNEMO 惯性模块：始终开启的 3D 加速度计和 3D 陀螺仪
LSM6DSM，iNEMO 惯性模块：始终开启的 3D 加速度计和 3D 陀螺仪
LSM6DSL，iNEMO 惯性模块：始终开启的 3D 加速度计和 3D 陀螺仪
AN4844，应用笔记，LSM6DS3H：始终开启的 3D 加速度计和 3D 陀螺仪

版本历史

日期	版本	变更
2016 年 7 月 13 日	1	初始版本